

REMARKS

Claims 24 and 59-68 have been amended. No claims have been added or cancelled. Claims 1-68 are pending in the application. Claim 24 has been amended to correct a minor dependency issue. Reconsideration is respectfully requested in light of the following remarks.

Section 101 Rejection:

The Office Action rejected claims 59-68 under 35 U.S.C. § 101 as being directed to non-statutory subject matter. Claims 59-68 have been amended to recite a tangible computer accessible medium. Applicants respectfully request removal of this rejection.

Section 103(a) Rejection:

The Office Action rejected claims 1-5, 11-17 and 22-28, 34-37, 42-44, 48-52, 59, 63-65, and 68 under 35 U.S.C. § 103(a) as being unpatentable over Bittinger et al. (U.S. Patent 6,453,362) (hereinafter "Bittinger") in view of Winer, "XML-RPC for Newbies" (hereinafter "Winer"). Applicants respectfully traverse this rejection for at least the reasons presented below.

Regarding claim 1, contrary to the Examiner's assertion, Bittinger in view of Winer fails to teach or suggest that the message in the data representation language further includes a credential for allowing the client access to a service configured to perform functions on behalf of clients in the distributed computing environment. Bittinger teaches the use of a ticket object created by the client that a server application can use to pass a server stub object to the client. The client can then use the server stub object to invoke remote methods on the server application (Bittinger, column 6, lines 63-67, column 7, lines 42-47, and column 8, lines 15-22). The Examiner argues that Bittinger's login procedures and ticket object correspond to a client including in a data

representation language message a credential allowing the client access to a service. However, under Bittinger, the ticket is used to notify the client that the server application is running and ready to receive requests and provides a method for the server application to provide a server stub object to the client. Additionally, Bittinger teaches that *only a ticket identifier*, and not the actual ticket, is included in the message sent by a client to request that an application be started on the server (Bittinger, column 3, lines 26-28 and lines 47-52). The ticket in Bittinger is not a credential, included in a data representation language message that also includes information representing a computer programming language method call, for allowing the client access to a service configured to perform functions on behalf of clients in the distributed computing environment.

Further, Bittinger teaches the use of a separate authentication server to authenticate the client using well-known login procedures such as user id and password. After authenticating the client, the authentication server then forwards the ticket to the server application (Bittinger, column 7, line 67 – column 8, line 8). The ticket in Bittinger is not in a message with a representation of a method call for a function of a service. Therefore, Bittinger does not teach the use of a ticket *as a credential* as suggested by the Examiner. Furthermore, the login procedure referred to by the Examiner has nothing to do with including a credential with a representation of a method call for a function of a service. Moreover, Bittinger clearly does not teach a single message that includes both a representation of a computer programming language method call and a credential for allowing the client access to a service. Winer is not concerned with, nor teaches anything regarding, including a credential in a message and thus fails to overcome this deficiency of Bittinger.

Bittinger in view of Winer also fails to teach or suggest the service examining the credential included in the message and performing a function on behalf of the client in accordance with the information representing the computer language method call included in the message if the credential is authentic. In contrast, Bittinger teaches the use of a separate server authenticating a client *prior to the launching* of the desired server application (Bittinger, column 3, lines 45-64 and column 8, lines 9-14). Bittinger fails to

teach that any credential is verified by the service application that performs the function in accordance with a representation of the method call. In addition, Bittinger teaches that the application server only uses the received ticket identifier to obtain a client stub from a client repository in order to invoke an acknowledgement method of the client stub (Bittinger, column 3, lines 56-64).

As noted above, Winer does not teach anything regarding a server examining a credential included in a message than thus fails to overcome the above noted deficiency of Bittinger. Therefore, Bittinger in view of Winer clearly does not disclose the service examining a credential included in the same message that includes a representation of the computer programming language method call.

For at least the reasons above, the rejection of claim 1 is not supported by the prior art and removal thereof is respectfully requested. Furthermore, the rejection of claims 25, 45, 49, and 59 is unsupported by the prior art for similar reasons as discussed above.

Regarding claim 5, Bittinger in view of Winer fails to teach or suggest the client message endpoint attaching the credential to the message. The Examiner cites column 7, lines 1-5 of Bittinger and argues, "tStamp is an identifier used on all messages." The Examiner's interpretation of Bittinger is incorrect. The Examiner argues that Bittinger's tStamp is equivalent to a credential attached to a data representation language message. However, Bittinger teaches, "the identifier 'tstamp' may be used *to locate the ticket* within the [client-side registry] database" (emphasis added, Bittinger, column 7, lines 8-9, see also column 3, lines 47-64). Bittinger further describes a server using the tstamp to retrieve a ticket stub from the client-side registry (Bittinger, column 7, lines 41-43). After retrieving the ticket stub, the server invokes an acknowledgement method of the ticket stub. Thus, Bittinger's tStamp cannot be considered any sort of credential. Nor does Bittinger teach that a tstamp is used on all messages, contrary to the Examiner's assertion. Bittinger's use of a tstamp to allow a server application to locate and retrieve a ticket stub from a client-side registry clearly does not constitute a client message

endpoint attaching the credential to a data representation language message. Winer fails to mention anything regarding a client message endpoint attaching a credential to a message, nor does the Examiner rely upon Winer. Thus, the combination of Bittinger and Winer fails to teach or suggest the client message endpoint attaching the credential to the message.

For at least the reasons above, the rejection of claim 5 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments apply to claim 28.

Regarding claim 15, Bittinger in view of Winer does not teach or suggest a method wherein a service includes a service method gate configured to provide an interface to the one or more computer programming language methods of the service by receiving data representation language messages and invoking computer programming language methods specified by the messages. The Examiner cites column 7, lines 41-57 of Bittinger and argues that Bittinger's server stub "is used as a gate to provide an interface to one or more computer programming language methods." However, the server stub in Bittinger does not provide an interface to computer programming language methods of a service *by receiving data representation language messages*. Instead, the server stub in Bittinger allows a client to *invoke* methods of the server stub. The purpose of Bittinger's server stub (as well as all RMI stub objects) is to allow a client to interact with the server stub as if it were the remote server object. Nowhere does Bittinger mention a server stub receiving data representation language messages. Instead, Bittinger's server stub is received by the client from the server-side registry and the client directly invokes methods of the server stub (Bittinger, column 6, lines 28-31). Bittinger's server stub never receives any data representation language messages nor does the server stub invoke any computer programming language methods specified by any messages.

Winer is not relied upon by the Examiner nor does Winer overcome the above noted deficiency of Bittinger. Thus, Bittinger and Winer, both singly and in the Examiner's proposed combination, fail to teach or suggest anything about a service

including a service method gate configured to provide an interface to the one or more computer programming language methods of the service by receiving data representation language messages and invoking computer programming language methods specified by the messages.

For at least the reasons above, the rejection of claim 15 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments apply to claims 36 and 65.

The Office Action rejected claims 18-20, 38-40, 47, 53 and 66 under 35 U.S.C. § 103(a) as being unpatentable over Bittinger in view of Leach et al. (U.S. Patent 6,108,715) (hereinafter "Leach"). Applicants respectfully traverse this rejection in light of the following remarks.

First of all, the rejection of claims 18-20, 38-44, 47, 53 and 66 is improper because the Examiner has rejected them over Bittinger in view of Leach, but relies upon a different prior art reference (Winer) for the rejection of their respective independent claims.

Furthermore, in regard to claim 18, Bittinger in view of Leach (and Winer) does not teach or suggest storing the generated results data to a space service in the distributed computing environment, providing an advertisement for the stored results data to the client, wherein the advertisement comprises information to enable access by the client to the stored results data, and the client accessing the stored results data from the space service in accordance with the information in the provided advertisement. The Examiner states, "Leach discloses the creation of a data stack used to store the results of the operations locally on the server, then allows the client to map back to the stored results stack by providing an address which allows direct transfer between the client and the server greatly reducing the processing overhead." Applicants fail to see the relevance of

Leach's teachings to the limitations of claim 18. Furthermore, Applicants assert that the Examiner's interpretation of Leach is incorrect.

Leach teaches direct stack-to-stack transfer of parameters between two processes executing concurrently *on a single machine*. Rather than marshaling and unmarshaling parameters when invoking a remote procedure call in a second process, Leach teaches the use of a shared data stack and that a operating system kernel process copies the input and output parameters between the two processes individual stacks (Leach, column 4, lines 32-43). Additionally, Leach teaches directly copying data from a server process to a client process through the use of a kernel that can access the address spaces of both processes, thereby facilitating such a direct transfer of data (Leach, column 5, lines 44-52). Thus, contrary to the Examiner's assertion, Leach does not disclose the creation of a data stack used to store the results of the operations locally on the server and then allows the remote client to map back to the stored results stack by providing an address.

Specifically, under Leach, the client is not provided an address to map back to any stored results stack, as the Examiner contends. Instead, Leach teaches that a kernel process directly transfers the output parameters onto the client's stack and "transfers control to the client process with the instruction pointer pointing to the instruction that follows the client process' call to the proxy method" (Leach, column 5, lines 52-55). Hence, Leach's kernel transfers any results directly into the client's stack and does not provide the client any address or information that would allow the client to "map back" to the stored results stack. Further, Leach teaches that the kernel must do this, because the client would not be able to access the other process' stack. Thus Leach **teaches away from** enabling access by the client to the stored results data (Leach, column 6, lines 51-54).

Bittinger in view of Leach (and Winer) clearly fails to teach or suggest storing the generated results data to a space service in the distributed computing environment, providing an advertisement for the stored results data to the client, wherein the advertisement comprises information to enable access by the client to the stored results

data, and the client accessing the stored results data from the space service in accordance with the information in the provided advertisement. Instead, the combination of Bittinger and Leach (and Winer) proposed by the Examiner would only result in a system in which a client uses Bittinger's server stub to invoke methods of a remote server application and wherein any results from the method would be directly transferred to the client's stack by a kernel capable of directly accessing the client's stack, as taught by Leach.

Therefore, for at least the reasons above the rejection of claim 18 is not supported by the prior art and removal thereof is respectfully requested. Similar remarks as those above regarding claim 18 also apply to claim 38.

Regarding claim 19, Bittinger in view of Leach (and Winer) fails to teach or suggest generating a client results message endpoint in accordance with the information in the provided advertisement, wherein the client results message endpoint is configured to send messages in the data representation language to the space service for the client; generating a results request message in the data representation language, wherein the results request message requests the results data be provided to the client; the client results message endpoint sending the results request message to the space service; and the space service sending the requested results data to the client results message endpoint in response to the results request message. The Examiner does not cite any portion of Bittinger or Leach that mentions anything about a space service or about the generation and use of either a client results message endpoint or results request messages.

The Examiner cites only Leach's teachings regarding a kernel that copies output data resulting from a remote procedure call directly from the remote procedure's stack to the client's stack. However, as noted above regarding claim 18, Leach teaches that the kernel copies the output data after capturing processing control when the remote procedure has completed its execution (Leach, column 3, lines 39-41). Leach does not mention anything about generating a client results message endpoint, nor does Leach describe generating a results request message in a data representation language. In fact, **Leach teaches away** from generating a client results message endpoint and generating a

results request message in a data representation language by teaching the automatic and direct transfer of output data from the remote procedure's stack to the client stack, as discussed in more detail above regarding claim 18. Once the output data is in the client's stack, the client has direct and complete access to them. There is no need, nor does it make sense, to generate a results message endpoint or to generate results request messages to obtain results data when the results data are already readily available in the client stack.

For at least the reasons above the rejection of claim 19 is not supported by the prior art and removal thereof is respectfully requested. Similar remarks as those above regarding claim 19 also apply to claim 39.

Regarding claim 20, Bittinger in view of Leach (and Winer) fails to teach or suggest wherein the information (in the advertisement for the stored results data) to enable access by the client to the stored results data comprises one or more Uniform Resource Identifiers (URIs) for accessing the stored results data. The Examiner contends, without citing any passage or portion of Leach, that Leach teaches the use of an address for access a data stack and that "the types of files are known by the client, which are key features of a URI". The Examiner is presumably referring to Leaches teachings regarding directly transferring results data from a remote procedures address space to a clients stack. However, Leach teaches nothing regarding Uniform Resource Identifiers or anything that could be considered a URI. The only addresses Leach refers to are memory address in virtual memory address spaces within a single computer system.

Furthermore, Leach does not mention anything regarding types of files that are known by a client, as the Examiner asserts. As described above, Leach teaches a method for automatically and directly transferring output data from a remote procedure's address space to a client's stack, thereby giving the client ready and easy access to the output data. Leach fails to teach anything regarding an advertisement for stored results data and further fails to disclose that such an advertisement comprises information to enable

access by the client to the stored results data that includes one or more URIs. Additionally, there is no need for a client to use URIs to access stored results data because Leach copies output data to the client's stack, thereby providing the client direct access to the output data. Furthermore, one of ordinary skill in the art would not use a URI for a client to access data stored on the client's own stack as in the system of Leach.

Therefore, the Examiner's proposed combination of Bittinger and Leach (and Winer) clearly fails to teach or suggest wherein the information (in the advertisement for the stored results data) to enable access by the client to the stored results data comprises one or more Uniform Resource Identifiers (URIs) for accessing the stored results data. Thus, for at least the reasons above, the rejection of claim 20 is not supported by the prior art and removal thereof is respectfully requested. Similar remarks as those above also apply to claim 40.

The Office Action rejected claims 21, 41 and 67 under 35 U.S.C. § 103(a) as being unpatentable over Bittinger in view of the Instaweb Online Computing Dictionary (Instaweb, <http://www.instantweb.com/foldoc/foldoc.cgi?query=XML>) (hereinafter "Instaweb"). Applicants respectfully traverse this rejection for at least the following reasons.

Claims 21, 41 and 67 are patentable for at least the reasons given above regarding their respective independent claims. Furthermore, the rejection of claims 21, 41 and 67 is improper because the Examiner has rejected them over Bittinger in view of Instaweb, but relies upon a different prior art reference (Winer) for the rejection of their respective independent claims.

Additionally, regarding claim 21, the Examiner states, "XML can be used to create custom tags for data objects that offer greater flexibility in organizing and presenting information." However, applicants' claim 21 is not concerned with data objects or in organizing or presenting information. Neither is Bittinger. Bittinger teaches

a method for remote invocation of server applications. Specifically, Bittinger teaches a client login and registration process to coordinate the launching of a server application and relies upon well known, prior-art, remote method invocations between clients and server applications (*See*, Bittinger, column 5, line 55-column 6, line 28).

Furthermore, the combination of Bittinger and InstaWeb would not result the method recited by applicants' claim 21. InstaWeb teaches only that XML can be used to create custom tags for data objects that offer greater flexibility in organizing and presenting information. InstaWeb does not suggest that XML be used for generating messages that include information representing computer programming language method calls or that include credentials for allowing a client access to a service configured to perform functions on behalf of clients, as recited in applicants' claim 21. Bittinger relies upon remote invocation methods such as RMI and CORBA that do not use data representation languages. Any combination of Bittinger and InstaWeb would still use prior-art remote invocations such as RMI and CORBA (which do not use data representation language messaging). The Examiner's proposed combination would result only in a system that uses Bittinger's system for launching a server application, but that also uses XML for organizing and presenting information, as taught by InstaWeb. For at least the reasons above, the rejection of claim 21 is not supported by the prior art and removal thereof is respectfully requested. Similar remarks apply to claims 41 and 76.

The Examiner rejected 60 - 62 for the same reasons cited for claims 6, and 7, respectively. However the Examiner also indicated that claims 60 - 62 would be allowable if rewritten in independent form. Therefore, the rejection of claims 60-62 is clearly improper and removal thereof is respectfully requested.

In regard to all of the rejections, Applicants also assert that numerous ones of the dependent claims recite further distinctions over the cited art. However, since the rejection has been shown to be unsupported for the independent claims, a further discussion of the dependent claims is not necessary at this time.

Allowable Subject Matter:

Claims 6-10, 29-33 and 60-62 were objected to as being dependent upon a rejected base claim, but otherwise allowable if rewritten in independent form. In light of the above remarks, Applicants assert that claims 6-10, 29-33 and 60-62 are allowable as depending from patentably distinct base claims. Applicants therefore respectfully request allowance of claims 6-10, 29-33 and 60-62 as currently pending.

Allowed Claims:

Claims 55-58 have been allowed.

CONCLUSION

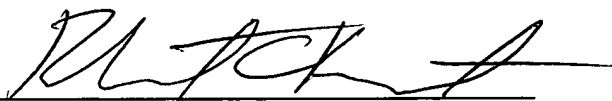
Applicants submit the application is in condition for allowance, and notice to that effect is respectfully requested.

If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5181-67300/RCK.

Also enclosed herewith are the following items:

- ☒ Return Receipt Postcard
- ☐ Petition for Extension of Time
- ☐ Notice of Change of Address
- ☐ Fee Authorization Form authorizing a deposit account debit in the amount of \$
for fees ().
- ☐

Respectfully submitted,



Robert C. Kowert
Reg. No. 39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: June 24, 2005